

Title of the Prior Art

Japanese Published Patent Application No. Sho.59-58580

Date of Publication : April 4, 1984

Inventor : Takayuki Nakagawa et al.

Concise Statement of Relevancy

This reference discloses a masking vector arithmetic processor, in which generation of a vector mask and a conditional vector arithmetic processing are simultaneously carried out in parallel with each other. However, the vector arithmetic processor described in this reference comprises plural registers specifically for vector masks, and generates the vector mask by a circuit of a system which is different from a system for obtaining arithmetic processing target data. This vector arithmetic processor does not decide whether arithmetic is to be executed or not, in parallel with the process for obtaining the arithmetic processing target data, as the present invention defined in Claims 1 and 3. Further, this vector arithmetic processor does not decide the state of supplied source data in the pipeline, as the present invention defined in Claims 2 and 4.

According to Claims 1 and 3 of the present invention, when the arithmetic processing target data is obtained in an arithmetic processing step of executing an arithmetic processing, it is computed and decided whether the arithmetic is to be executed or not, in parallel with the process of obtaining the target data. Thereby, a pipeline processing of performing a conditional arithmetic of whether an arithmetic using input data is to be executed or not, without using a specific mask register, and retaining the result can be executed without interrupting the flow.

This Page Blank (uspto)

According to Claims 2 and 4 of the present invention, the state of supplied source data is decided in the pipeline, and the conditional arithmetic is performed depending on the state of the source data as the condition. Thereby, a pipeline processing of performing a conditional arithmetic of whether an arithmetic using the source data is executed or not, without using a specific mask register, and retaining the result can be executed without interrupting the flow. In addition, the overhead in which the condition for performing the conditional vector arithmetic is computed can be dispensed with.

This Page Blank (uspto)

⑨ 日本国特許庁 (JP)

⑪ 特許出願公開

⑫ 公開特許公報 (A)

昭59—58580

⑬ Int. Cl.³
G 06 F 15/347

識別記号

庁内整理番号
7056—5B

⑭ 公開 昭和59年(1984)4月4日

発明の数 1
審査請求 未請求

(全 8 頁)

⑮ マスク付きベクトル演算処理装置

地株式会社日立製作所中央研究
所内

⑯ 特 願 昭57—168358

⑰ 発 明 者 阿部仁

⑱ 出 願 昭57(1982)9月29日

秦野市堀山下1番地株式会社日
立製作所神奈川工場内

⑲ 発 明 者 中川貴之

⑳ 出 願 人 株式会社日立製作所
東京都千代田区丸の内1丁目5
番1号

国分寺市東恋ヶ窪1丁目280番
地株式会社日立製作所中央研究
所内

㉑ 発 明 者 長島重夫

㉒ 代 理 人 弁理士 薄田利幸

国分寺市東恋ヶ窪1丁目280番

明 細 書

発明の名称 マスク付きベクトル演算処理装置
特許請求の範囲

1. 2つ以上のベクトルマスク専用レジスタと、
少なくとも1つのベクトルマスク専用演算器と、
該ベクトルマスク専用レジスタへの書き込み処理
に並行して、同一のベクトルマスク専用レジス
タからの読出し処理を行うための同期の手段と、
ベクトルマスク専用レジスタ間での演算命令の
処理にあたって、要求されるマスク情報間での
読み出し及び演算の同期手段と、マスク付きベ
クトル演算命令の処理にあたって、要求される
ベクトルデータ及びマスク情報の間での読出し
及び演算処理の同期手段とを有する、マスク付
きベクトル演算処理装置。

2. 該ベクトルマスク専用レジスタのうち、同一
のものに対する書き込み処理と読出し処理の間で
の該同期手段は、書き込み命令に後読する、読出
し命令の処理における、ベクトルエレメント毎
の同期をとる手段を有する第1項の装置。

3. 該マスク付きベクトル演算命令において、該
ベクトルデータの保持及び読出しに、ベクトル
レジスタを有する第1項の装置。

4. 該マスク付きベクトル演算命令において、ベ
クトルレジスタ中のデータに対し、マスク付き
の主記憶格納内の手段を有する第3項の装置。

5. 該マスク付きベクトル演算命令以外の演算命
令の解説結果として選択する演算器として、マ
スク付き演算命令に用いる演算器を流用する手
段を、ベクトルデータとベクトルマスクデータ
との該同期手段と、ベクトルマスクデータの読
出し手段の一部として有する第1項の装置。

発明の詳細な説明

[発明の利用分野]

本発明はベクトルデータを高速に演算する装置
に関する。

[従来技術]

ベクトルプロセッサのより高速な処理を達成する
には、より多くの種類の処理をベクトルプロセサ
により高速に処理可能とすることが課題となつて

いる。中でも、FORTRAN プログラムで、IF 文を含む DO ループを高速に処理するには、より高度の処理装置が必要とされるが、第1図に示すような DO ループの従来技術による処理手順を第2図と第3図を用いて説明する。

第1図に示す DO ループでは、式(1)にあるような論理演算の結果の値が“真”であり、かつ、式(2)に示す論理演算結果が“真”である場合に限り対応するインデクス値：I を持つエレメント間で式(3)の加算及び代入を行い、式(1)か、式(2)のいずれかが成立しないようなインデクス値を持つエレメント間での式(3)の加算及び代入は行わない。このような処理をN組のエレメントについて繰り返す。

第2図は、エレメント数：Nが6で、オペランドA(1~6)、B(1~6)、C(1~6)、D(1~6)、F(1~6)、G(1~6)に適當な数値を仮定し、第1図の演算を行なった場合のデータの流れを示したものである。

第2図で値“X”はこれらの処理により変更を

受けず、また、この処理に関与しないことを示す。

以下に、その処理手順を示す。

ステップ1： オペランドA(1~6)とB(1~6)の対応するエレメント同士を、それぞれ比較してベクトルマスク：VM(1~6)を作成する。この場合、対応する両エレメントの値が一致して論理演算結果が“真”となるとき、VMには値“1”を書き、一致しないときVMには値“0”を書くものとする。従つて本例ではVM(1~6)の値はそれぞれ、0、1、0、1、1、0となる。

ステップ2： 次のステップ3で、ベクトルマスク：VMを格納している、ベクトルマスクレジスタを、再度書き込みに使用する為に、VM(1~6)の内容を、別のレジスタSR1に送渡させる。なお、このSR1は、ベクトルマスク専用のレジスタではなく、保持データの全エレメント分を一括して処理することしか出来ない。その為に、このステップは、ステップ1が全て終了してから行われる。

ステップ3： ステップ(1)と同様に第1図の式(2)の演算を行い、結果をVM(1~6)に書き込む。ここでも、論理演算値“真”に対して値“1”、“偽”に対して値“0”を対応させると、この場合のVM(1~6)の値は、それぞれ0、1、1、1、0、1となる。

ステップ4： 次のステップ5で、全エレメント分の一括処理しか出来ない汎用の演算器を用いて汎用のレジスタ間の論理演算を行うため、VM(1~6)の値を汎用のレジスタSR2に送渡する。このステップもステップ2と同じ理由で、ステップ3が全て終了してから行われる。

ステップ5： SR1とSR2のビット毎の論理積をSR3に求める。このステップはステップ4が終了してから行われる。

ステップ6： ステップ5で求めたSR3の値をVM(1~6)に転送する。

ステップ7： ステップ6で得られたVM(1~6)の値をもとに、第1図の式(3)に示す加算と代入を行う。その際、対応するベクトルマスク

VM(I)の値が“0”のエレメントについては演算の結果を無効とする。すなわち、その時のE(I)の値を変更しない。本例では、第2及び第4エレメントについてのみ演算結果が主記憶上のE(I)の値を変更するように動作する。

以上の処理の様子を、第3図のタイムチャートで示す。タイムチャートの縦軸は各ステップを示し、横軸は時間を示す。各番号は、そのエレメント番号の処理の最初のサイクルを示しており、ステップ間の時間的ずれは、起動時間のずれを示している。

以上に述べてきたように、従来技術による複雑な条件文を含むDOループのベクトルプロセサでの処理は、ベクトルマスクレジスタが1つしかない為、一括処理用のレジスタに移動して、一括処理による演算処理を行い、出来あがつた最終的なベクトルマスクを、再びベクトルマスクレジスタに転送しなければ、条件付きの演算処理が出来ない。従つて、第3図に見るように、従来技術ではベクトル処理が、ステップ1と、ステップ3、ス

ステップ7に、それぞれ処理時間帯を分けられ、これらのステップ間では、処理の並列・高速化ができないという問題があつた。

一般に、ベクトルプロセサでは複数ベクトル演算の異なるエレメントについての処理を、同時並行に処理することで高速性を達成しているが、ステップ2, 4, 5, 6の処理がベクトルプロセサ向きでない為に、第4図のように、ステップ1, 3, 7を並列に処理することを妨げている。

〔発明の目的〕

本発明の目的は、ベクトルマスクの生成、ベクトルマスク間の演算処理、及び条件付きベクトル演算処理を、全て同時並行して処理可能とするベクトル演算処理装置を提供することにある。

〔発明の概要〕

このため、本発明による装置では、

- (1) 同時に書き込み・読出し可能な複数のベクトルマスク専用レジスタ。
- (2) ベクトルマスク専用レジスタ中のデータを入力として新しいベクトルマスクの値を算出する、

の部分の説明は簡単にするに止める。

第5図において、101は主記憶装置、102は主記憶制御装置、103はスカラ処理装置であり、104はその一部であるスカラ命令制御装置、105はベクトル命令制御装置である。

スカラ命令制御装置104では主記憶制御装置102と信号線201を介して主記憶装置101から順次読出した命令を解説し、これが通常のスカラ命令であるときは、スカラ処理装置103にて通常のスカラ演算処理を行い、信号線202を介し結果を主記憶装置101に書き込む。スカラ命令制御装置104で解説した命令が、ベクトル命令列の起動を指示する命令であれば、信号線203を介して、ベクトル命令列の主記憶装置101上での先頭アドレスと、処理すべきベクトルエレメント数と、起動信号をベクトル命令制御装置105へ渡す。引続き、ベクトル命令制御装置105は与えられたアドレスに従い、信号線201, 204を介してベクトル命令列を主記憶装置101から順次読出し、ベクトル命令を解説し、命令中で指

1個以上のベクトルマスク専用演算器。

- (3) ベクトルマスクの書き込みと、ベクトルマスク専用演算器への読出しを同期させる手段。
- (4) ベクトルマスクの書き込みと、マスク付きベクトル演算処理用演算器への読出しを同期させる手段。

とを、新たに設けることにより、ベクトルマスクの生成、ベクトルマスク間演算、条件付きベクトル演算処理を、全て同時並行して処理することとを可能としたものである。

ただし、本明細書では“演算処理”という言葉により、主記憶参照、ベクトル間の加減算等の演算、結果の主記憶への格納の全てのものをさし、本明細書の実施例ではマスク付き主記憶格納を例にとる。

〔発明の実施例〕

以下、本発明を実施例を参照して詳細に説明する。第5図は本発明の一実施例を示す。

本発明に直接関連しない装置部分は、通常のベクトルプロセサと同じ構成を有するものとし、セ

定された、レジスタ、演算器、メモリリクエタ等が使用可能な状態にあると判断した命令から、それぞれ信号線205, 206, 207を介して命令に必要なリソースを起動すると共に、同時に、処理すべきベクトルエレメント数を含めた制御情報を転送する。ベクトル命令のエレメント個々に対する処理は、ベクトルレジスタ制御装置110や、ベクトルマスクレジスタ制御装置120の送出する。1エレメント毎の処理の許可信号に従って進められる。以下、第1図に示したD0ループの処理を行う場合を例にとり、本発明によるベクトル命令の実行を説明する。本例では、ベクトルレジスタと呼ぶバッファ記憶を用いており、それぞれNエレメントからなる一連のデータ、A(1~N), B(1~N), C(1~N), D(1~N), E(1~N), G(1~N)を信号線211~216を介し、ベクトルレジスタ111~116にそれぞれ格納することが必要であるが、この間の処理は通常のベクトルプロセサによる処理に従うものとする。

第1図の処理を実行するには、上記の処理を行う6つの命令の他に、A(I)とB(I)を要素ごとに比較し、比較結果をベクトルマスクレジスタに書き込む第7の命令。同じく、C(I)とD(I)の値の要素ごとの比較結果を第7の命令で書き込んだとは別のマスクレジスタに書き込む第8の命令。第7の命令と第8の命令により得られる2つのベクトルマスク間で論理演算を行い、結果を第3のベクトルマスクレジスタに書き込む第9の命令。F(I)とG(I)を要素毎に加算し、結果をベクトルレジスタに格納する第10の命令と、第10の命令によつて得られた加算結果を、第9の命令によつて得られたベクトルマスクが“1”をとる場合のみ(もしくは、“0”をとる場合のみ)、B(I)に相当するメモリアドレスに書き込む、第11の命令が用いられる。

以下第5図を用いて、第1図の処理を説明する。第7の命令に responding ベクトル命令制御装置は、演算器131に、ベクトルレジスタ111、112中の値を送り、式(1)の比較を行なう。この

場合、等しいという関係が成立する場合には、値“1”を、成立しない場合には値“0”を、信号線231を介し、ベクトルマスクレジスタ122に書き込むものとする。この比較演算は、131にパイプライン演算器を用いることで、1サイクルに1エレメントのビットで進めることができる。第8の命令は、同様に式(2)の比較を行い、成立に対し“1”を、不成立に対しては“0”を、パイプライン演算器132から信号線232を介して、ベクトルマスクレジスタ123に書き込む。

122と123に書き込まれた値の間で、今度は第9の命令によつて論理積をとる演算が行われる。これは、従来技術で述べたような、一括処理しか出来ないレジスタ及び演算器ではなく、ベクトルマスクの専用レジスタ121~123、及び専用のパイプライン演算器141を用いているため、必要なデータが揃ったエレメント間では、ただちに論理積をとる処理に移るよう同期される。

また、第9の命令で作成されたベクトルマスクの値を、第11の命令で参照する場合も、メモリ

に格納するデータと、ベクトルマスクが揃ったエレメント間では、直ちに処理が行われるよう同期される。

前者のベクトルマスクレジスタ間の同期機構を第6図で説明し、後者のベクトルマスクレジスタとベクトルレジスタの間での同期機構を、条件付きベクトル演算を例に、第7図で説明する。

第7、第8の命令で書き込まれたベクトルマスクレジスタの値は、書き込みが全エレメントにわたって終了していれば、どこから読んでも良いが、途中のエレメントを書き込んでいる途中なら、書き込む前の値を読むと、第1図の処理が正しく実行されない。この書き込み済の範囲を知る目的で、例えば、比較演算結果を書き込んでから、論理積をとるため読出されるベクトルマスクレジスタ122(以下VMR2と呼ぶ)に対応して、書き込み済みエレメント数が、どれだけ読出しエレメント数を上回っているかを答える、アップダウンカウンタ322と、書き込み中であるとか読出し中であるという状態を答えるレジスタ321を用意する。

VMR2から1エレメント読み出す為の許可信号404は、VMR2が書き込み中でない場合か、書き込み中だがこれから読み出そうとするエレメントが書き込み済である場合に発行される。カウンタ322は書き込みエレメント数から読出しエレメント数(もしくは読出し予定エレメント数)を引いた差を保持しているので、このカウンタの値が正であることを示す信号401、VMR2が書き込み中かつ、読出し中であることを示す信号402、VMR2が書き込み中ではない、単純な読み出し状態にあることを示す信号403に、AND回路323、OR回路324の論理をとることで、信号404を得ることができる。

また、VMR2とベクトルマスクレジスタ123(以下、VMR3と呼ぶ)の間で演算をして良いのは、VMR2も、VMR3も、どちらも1エレメント読み出す許可信号が揃っている場合に限る。1エレメントの演算許可信号406は、VMR2の1エレメント読出し許可信号404と、VMR3の1エレメント読出し許可信号405をAND

回路301を用いてANDすることにより得られる。この1エレメントの演算許可信号406によつてVMR2と3の脱出しアドレス302の値が1追加され、未処理エレメント数を表したカウンタ303の値が1でないなら、1減じて、次のエレメント処理へと進む。この論理演算はパイプライン演算器141の各ステージ、1411～1413を経て、ベクトルマスクレジスタ121（以下VMR1と呼ぶ）に書き込まれる。この際、書き込み側のアドレス304やアップダウンカウンタ312への加算は、306～307に示す同期フリップフロップを適当に追加することで、データの処理の同期がとられる。VMR1に書き込んだエレメントデータを、必要な場合に、書き込みと同時に読み出すには、アップダウンカウンタ312への+1動作が行われてから、実際にデータが122へ書き込まれるまでの時間差を、アップダウンカウンタ312の-1動作が行われてから、脱出しが行われるまでの時間差に等しく設計することが必要である。本例ではそのために、信号407

は307のフリップフロップ群をバイパスしてアップダウンカウンタ312へ送られている。この307のフリップフロップのうち最初のもの3071から、VMR1までの信号の伝達時間と、312から、第6図では省略されているVR1の脱出しアドレス（第7図503）までの伝達時間を等しくすることによつて、VMR1に書き込まれたデータを直ちに脱出することが可能となつている。

第6図で、WTと書かれているのは、それぞれのベクトルマスクレジスタへの書き込み命令により発行される信号である。本例では、第1図の式(1)の処理の始まりと共に、信号421により、VMR2の状態321を、書き込み状態にセットし、アップダウンカウンタ322を0にリセットする。そして、1エレメント毎の書き込み信号422により、書き込み数をカウントアップさせると共に信号423により、結果をVMR1に書き込む。そして、全エレメント分の処理の終了をカウンタ303の値が1であるという信号424により検出し、書き込み状態にあるという記録を311から、脱出

し状態にあるという記録を321～331から解除する。

以上の動作は、第6図で説明しているベクトルマスクレジスタ間演算によるVMR1の書き込み制御方式と同じため、第6図ではVMR2及びVMR3の書き込み側の制御回路は省略してある。

第1図の式(1)、式(2)が全エレメント処理されるのを待たず、式(1)と式(2)のベクトルマスク間演算命令が起動されると、INITと書かれた信号207により、まず、VMR1の状態311が書き込み状態にセットされると共に、アップダウンカウンタ312が0にリセットされるのは、先に述べた式(1)の処理と同様であるが、さらにVMR2及びVMR3の状態か、読み出し状態にセットされる。従つて本例で第1図のプログラムの処理を行つた場合、VMR2は読み出しかつ書き込み中の状態となり、信号401がオンで信号403がオフのため、書き込み済のエレメントのみについて読み出しの許可を与える制御を行う。

前述の式(1)の命令により1エレメントの書き込み

みがある度に、カウンタ322は+1されるから、その結果を使うベクトルマスク間演算では、カウンタ322の値が正である限り、読み出し許可信号404を発行し、カウンタ322から-1する。例では、簡単のために408により-1しているが、406により-1してもよい。

この408により-1した場合は406により-1した場合に比べ性能の劣化する場合があるが、この劣化は391～394にある回路により回避できる。信号406により-1する場合391～394の回路は不要である。以下では、簡単のため、ベクトルレジスタ、ベクトルマスクレジスタの脱出し許可信号作成及び、演算許可信号作成時の391～394に相当する回路は図面中では省略する。

VMR2と同様にVMR3も、読み出し許可信号を作成し、両者のAND条件をとつた406が最終的な演算許可信号となつて読み出しアドレス302の値を進め、読み出し用セレクタ326、336を介して、ベクトルマスク専用演算器305

にデータを送ると共に、書き込み許可信号としてカウンタ312の値に+1を行い、書き込みアドレス304の値を進める。ここで、演算器141をベクトルマスクレジスタ間演算専用とすることで、僅かなコストの追加により、処理の高速化が可能となつている。第5図では、ベクトルレジスタ用演算器3個、ベクトルマスク専用演算器1個の構成となつているが、ベクトルマスク専用演算器は入力データ巾が、前者の32~64ビット/エレメントに対し、後者は1ビット/エレメントであり、コストもそれに順じて少なく出来る。

第6図の制御により、第7~8の命令と第9の命令が同期されることが判るが、第7図は、第11の命令と第12の命令の同期方法を示したものである。VR2とVR3の読み出し制御回路は、第6図のベクトルマスクレジスタのVMR2とVMR3のそれと同様のものを採用している為、ここでは説明を省く。VMR1の読み出し制御回路と第6図の読み出し制御回路との相違は、502のフリップフロップを条件付き命令でないことを

記録させるよう、新たに設け、その場合、演算器等に送るベクトルマスク値603を、信号602を介して常に"1"にしていることである。このフリップフロップにより、条件付きベクトル演算命令以外にも、同じ演算器505を使用できる。

また、第6図でVMR2とVMR3の脱出し許可信号間のANDをとつた信号406に相当して、第7図での演算処理許可(本例では1エレメントの主記憶格納許可)信号606は、ベクトルマスクレジスタVMR1の脱出し許可信号644と、ベクトルレジスタVR1の脱出し許可信号634と、主記憶書き込み制御回路1021からの受け付け許可信号601との間でANDをとつたものとしている。条件付き命令以外の主記憶書き込み命令の処理の場合は、信号207によりレジスタ502に1を格納し、信号602とOR回路543により、信号644を常に1に設定する。

〔発明の効果〕

以上のようにして、第4図のタイムチャートに示したように従来は3N+3サイクルかかつた処

理を、本発明によりN+3サイクルで処理できる。この短縮された2Nサイクルの内訳は、ベクトルマスク生成の比較演算命令と、ベクトルマスク間の論理積演算命令との間での処理の非並列化による従来方式での損失がNサイクル、論理積演算命令と、条件付きベクトル演算処理(本例では条件付き主記憶格納)命令との間での処理の非並列化による従来方式での損失がNサイクルであり、両損失の改善には、第6図と第7図に示した2通りの同期機構がそれぞれ寄与している。両改善とも、専用のベクトルマスクレジスタと演算器を使用せずに、通常のベクトルレジスタを拡張して使用しても可能であるが、専用化することにより、同じ性能を得る上で、コスト的に32~64倍有利であることが明らかである。

図面の簡単な説明

第1図はマスク付きベクトル処理を含むFORTRAN プログラム例。第2図は従来技術による第1図プログラムの処理手順。第3図は、従来技術による図1の処理のタイムチャート。第4

図は本発明による図1の処理のタイムチャート。

第5図は、本発明による処理装置の一構成例を示す。第6図は、ベクトルマスクレジスタ間の演算実行制御の一実施例、第7図は条件付きベクトル主記憶格納処理実行制御の一実施例を示す。121~123はベクトルマスク専用レジスタ。

141はベクトルマスク専用演算器。

代理人 弁理士 薄田利幸



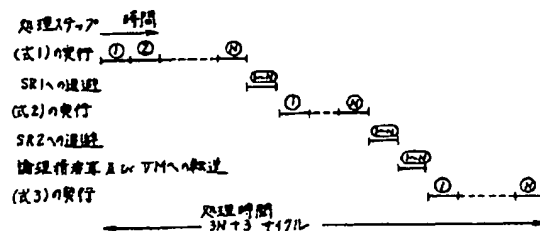
第 1 図

```

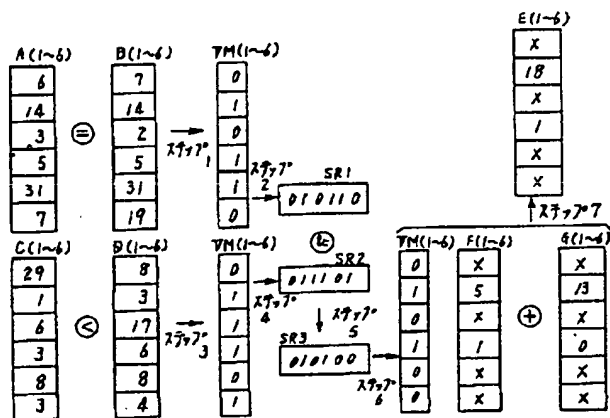
DO 10 I=1,N
  IF (
    R (A(I), EQ, B(I)) 式(1)
    R .AND.
    R (C(I), LT, D(I)) 式(2)
    R )
    R E(I) = F(I) + G(I) 式(3)
10 CONTINUE

```

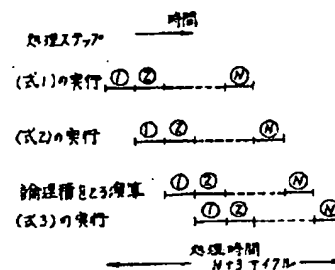
第 3 図



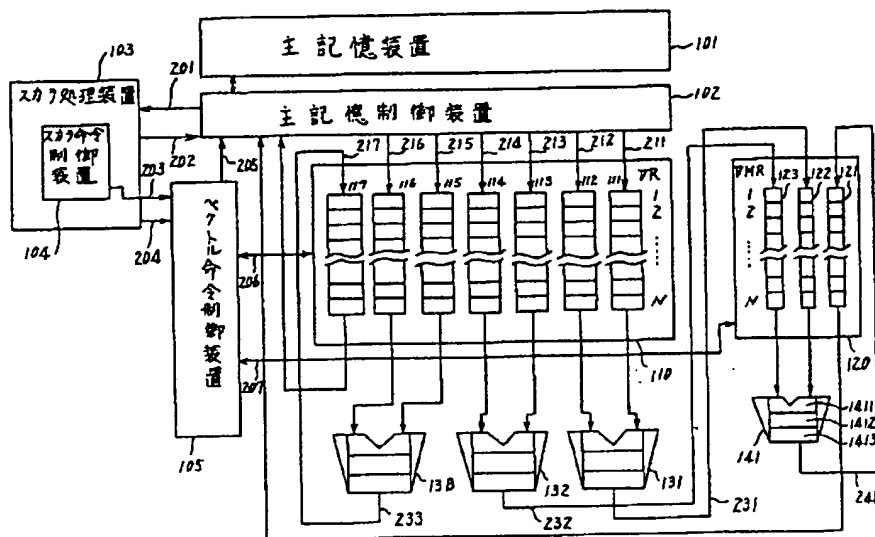
第 2 図



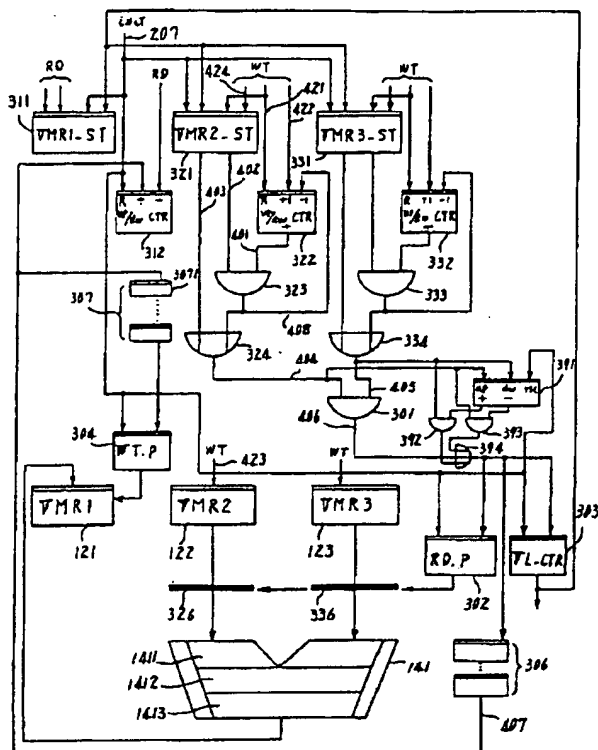
第 4 図



第 5 図



第 6 図



第 7 図

